

A Real-Time Deformable Detector

Karim Ali, François Fleuret, David Hasler, and Pascal Fua, *Senior Member, IEEE*

Abstract—We propose a new learning strategy for object detection. The proposed scheme forgoes the need to train a collection of detectors dedicated to homogeneous families of poses, and instead learns a single classifier that has the inherent ability to deform based on the signal of interest. We train a detector with a standard AdaBoost procedure by using combinations of pose-indexed features and pose estimators. This allows the learning process to select and combine various estimates of the pose with features able to compensate for variations in pose without the need to label data for training or explore the pose space in testing. We validate our framework on three types of data: hand video sequences, aerial images of cars as well as face images. We compare our method to a standard boosting framework, with access to the same ground truth, and show a reduction in the false alarm rate of up to an order of magnitude. Where possible, we compare our method to the state-of-the art, which requires pose annotations of the training data, and demonstrate comparable performance.

Index Terms—Image Processing and Computer Vision, Machine Learning, Object Detection.

1 INTRODUCTION

SUCCESSFUL techniques for object detection are based on machine learning. Though progress has been made in reliably detecting objects with a single pose, handling complex cases where object appearance is altered by viewpoint changes or deformations, has proven more difficult. This paper describes a framework which makes headway toward detecting objects regardless of their pose. We specifically address three types of pose variations: deformations, in-plane rotations and a limited range of out-of-plane rotation.

There are a number of recent works in literature proposing methods for dealing with pose variations. One common thread among most these works is that a collection of detectors, each trained for a single pose, is craftily combined in one form or another.

Some approaches [31], [35], [10] employ a two stage framework where pose is estimated as part of a first stage and a corresponding pose-specialized detector is tasked with classifying the image in the second stage. Other approaches [14], [13], [26], [27] proceed in a hierarchical fashion whereby pose estimation is gradually refined with classifiers dedicated to increasingly constrained poses. In all cases, training data must be annotated and partitioned into disjoint clusters, thereafter used to train a series of pose-dedicated classifiers.

Though reliable detection can be achieved in this manner, the underlying design of these methods raises an important difficulty: on the one hand, a fine partition of the pose space is clearly desirable to attain better detection performance while

on the other hand, finer partitions result in increased population size requirements. These techniques therefore compel a tradeoff between the granularity of the partition and the size of the training data. Equally troublesome is the fact that these approaches are burdened by the need to annotate data during training and by a more costly training. As a result, dealing with a fine partition of a rich pose space quickly becomes intractable using such a strategy.

Recently, the authors in [8] present a framework centered on *pose-indexed* features. The key idea revolves around analytically parameterizing the detector's constituent features with the pose. This avoids the need to partition the pose space and enables training to be carried out on the entire unfragmented data set. Nevertheless, the procedure still requires the data to be annotated for training while a search over the pose space is required for testing.

We propose a new approach which consists of treating pose as a collection of hidden variables and designing a family of pose estimators able to compute meaningful values for those variables directly from the signal. We allow the learning procedure to automatically handle the trade-offs involved in selecting and combining estimates of the hidden parameters obtained from various image areas. This approach sets forth a framework that overcomes both the data fragmentation problem, associated with the training of pose-dedicated classifiers, as well as the labeling and computational overheads of purely pose-indexed methods.

Our approach is a monolithic one in that a single classifier is built that can adaptively deform to detect a target. Our key contribution lies in augmenting a set of pose-indexed features with a *family of pose estimators*. Each feature then consists of a pair of functionals: one functional to estimate the pose and the other to compute a pose-indexed feature *parameterized* by the estimated pose. Various modes of parameterization are allowed each of which acts as a specific form of feature normalization. Though our framework is valid for any learning method, we rely here on the AdaBoost algorithm for its simplicity and efficiency [9], [32]. The AdaBoost learning

-
- K. Ali is with the École Polytechnique Fédérale de Lausanne (EPFL), Switzerland and with the Swiss Center for Electronic and Microtechnology (CSEM), Neuchâtel, Switzerland. E-mail: karim.ali@epfl.ch
 - F. Fleuret is with the Idiap Research Institute, Martigny, Switzerland, and the École Polytechnique Fédérale de Lausanne (EPFL), Switzerland. E-mail: francois.fleuret@idiap.ch
 - D. Hasler is with the Swiss Center for Electronic and Microtechnology (CSEM), Neuchâtel, Switzerland. E-mail: david.hasler@csem.ch
 - P. Fua is with the École Polytechnique Fédérale de Lausanne (EPFL), Switzerland. E-mail: pascal.fua@epfl.ch

procedure is allowed complete freedom in deciding how best to combine a pose estimator with a pose-indexed feature. In this manner, training proceeds on the unpartitioned data set while pose estimator learning and feature learning occur jointly in an integrated framework. The final detector consists of a variety of features which can deform independently based on the signal of interest, and on the pose variations observed in training.

This work was initially motivated by a practical application – the detection of hands to prevent injuries in manufacturing plants – which naturally poses significant challenges. The appearance of the hand, a deformable, articulate object may change considerably and to be of practical interest, detection must proceed in real-time with nearly zero error rates. We demonstrate that our framework provides substantial benefits in this setting. Moreover, we validate our framework on images of faces where pose variations consists essentially of rigid rotations and again show significant gains. Finally, we process aerial images of cars, characterized by in-plane rotation changes, and demonstrate gains of up to an order in magnitude. In all cases, the reference baseline is that of a standard boosting method with access to the same ground truth, namely data that is not annotated for pose. Whether faced with in-plane rotations, a limited range of out-of-plane rotations, or deformations, our framework readily adapts to the data and appears to sensibly combine the various pose estimates induced from training.

2 RELATED WORK

Tremendous progress has been made towards the reliable detection of objects in images. In particular, there is an extensive literature dealing with detecting objects under limited changes in view-angle, for instance frontal faces. Though algorithmic details vary greatly, works such as [32], [4], [19], [16], [20], [33] have been proven successful in unconstrained, cluttered or partially occluded scenes.

The problem of detecting objects regardless of their pose and where significant changes in appearance arise has proven more difficult. In its broadest definition, object pose includes all those latent variables which modulate object appearance such as location, scale, rigid rotations or view-angle changes, deformations, and variations in illumination. Works such as those described above and their extensions handle these pose parameters with various methods. Whereas variations in illumination may be dealt with at the feature level, by designing invariants such as edge detectors, location and scale are better handled via image normalization in training and exploration in testing: a classifier is trained for a single location and scale while detection is managed by searching for the presence of the target over all scales and locations of a given scene.

The predominant strategy, on the other hand, for dealing with view-angle changes and deformations consists of carefully combining a collection of classifiers each dedicated to a single pose. For example, the authors in [31] extend the Viola-Jones detector to address two types of pose variation concerning faces: in-plane rotations and out-of-plane rotations. To deal with in-plane rotations, the pose of the image of

interest is estimated using a decision tree constructed to determine the view class. Second, one of twelve rotation-specific Viola-Jones detectors is used to classify the image. The treatment of out-of-plane rotations is entirely analogous.

A number of other recent works essentially devise the same strategy in dealing with multi-view object detection [21], [35], [10], [18]. Multiple detectors, each specialized to a specific pose, are built and the pose is estimated as part of a first stage. Other works [14], [13], [26], [27] also employ pose dedicated classifiers with the notable difference that pose estimation and detection are organized hierarchically within a pyramid system. In these methods, each level of the pyramid gradually refines the pose estimate by the use of more constrained pose dedicated classifiers. Still, other works [24], [25], [30], [29] run a bank of pose dedicated classifiers on the scene and use various forms of arbitration logic to combine the output.

This difference in treatment when compared with the normalization and exploration strategy employed for location and scale stems from the fact that image normalization is not possible when faced with complex deformations or view-angle changes other than in-plane rotations. Hence, in the absence of a three dimensional model or in order to avoid the difficulties associated with building such a model, the view-based approaches described above are a sensible course of action and have been demonstrated to yield reliable detection performance. However, these techniques remain burdened by several difficulties. First and foremost, training data must be appropriately annotated in order for it to be partitioned into clusters of similar poses. Second, this partitioning or fragmentation of the available training data reduces the number of samples used to train each pose-dedicated classifier and negatively impacts performance. It is not difficult to conceive a setting where such a strategy fails to provide acceptable error rates: dealing with a rich pose space or a fine partition of the pose space, for instance, is indeed not possible using such a strategy without increasing training data size and training time.

In order to overcome training data fragmentation the authors in [8] present a framework centered on pose-indexed features. By allowing features to be parameterized with the pose, it becomes possible to treat in-plane rotation, ranges of out-plane-rotations and deformations in the same manner as location and scale are typically handled. All pose parameters are treated within the same formalism: pose-indexed features effect normalization during training while in testing, exhaustive pose exploration becomes necessary. Though promising results are shown, this technique requires nonetheless the training data to be labelled with the corresponding ground truth and incurs a significant computational cost in testing.

Also relevant are works such as [7], [2], [3], [5], [12], [17] which rely on sparse representations based on interest points. These approaches construct clusters of interest points, treated as object parts and spatially combined in a probabilistic fashion. This category of work has also proven successful in detecting objects with limited changes in view-angle. The use of sparse representations has been recently applied to the multi-view setting [11], [34], [23], [15], [28] with some success. Though the utilized points of interest effect pose estimation and normalization, these techniques fail to provide

acceptable error rates: at low to moderate image resolutions, an insufficient coverage of feature points leads to highly unreliable detection performance. Our approach bears some similarity to that of [6]. There, a view-based approach is combined with deformable parts. Whereas this method has proved successful in the multi-view setting it is nevertheless burdened by the need to explore possible configurations in testing. Also, much as the above works on sparse representations, this method fails to provide acceptable error rates at low image resolutions.

Our approach utilizes the pose-indexed features of [8] and requires neither labeling for rigid rotations and deformations, nor exploration of these pose parameters in testing. In contrast with the works on sparse representations, we do not rely on hand-designed local estimation and normalization. Instead, we introduce a family of pose estimators, which provide estimates of the rigid rotations and deformations from various areas in the image, and allow the learning procedure to choose the best combinations of pose-indexed features and pose estimators: thus a pose-indexed feature may obtain a pose estimate from one area in the image and compute a response in another. We also allow the learning procedure to select from several modes of normalization for each pose-indexed feature. The result is a flexible detector which weights dense features, each optimized with the best pose estimate and with the best normalization mode. As will be seen through our experiments and as shown in Figure 1, this permits the automatic discovery of the variations present in the training data while maintaining the generalization properties of the detector and providing reliable detection.

3 BACKGROUND

Formal presentations of both standard features and pose-indexed features are given here. In the remainder of this paper, we use the AdaBoost learning procedure to illustrate the various concepts. This is done for the sake of simplicity and because our implementation relies on such a setup. The underlying concepts, however, are not contingent on the use of a specific learning algorithm: one could indeed use pose-estimator based features in conjunction with other discriminative machine learning methods, such as Support Vector Machines and decision trees, or even with generative models.

3.1 Boosting with standard image features

Let $\mathcal{I} = [0, 1]^{W \times H}$, denote the space of gray scale images of size $W \times H$ and let

$$(X^{(i)}, Y^{(i)}) \in \mathcal{I} \times \{-1, 1\}, \quad i = 1, \dots, T, \quad (1)$$

denote a labelled training set where $i = 1, \dots, T$ is an index running through all available scenes. Here, we consider a *classification* setup so that the images $X^{(i)}$ either contain a target or not. Given a set \mathcal{H} of image features or mappings of the form

$$h_k : \mathcal{I} \rightarrow \mathbb{R}, \quad k = 1, \dots, K, \quad (2)$$

a standard AdaBoost procedure constructs a *strong* classifier f as a linear combination of, for instance, *stumps* of the following form

$$\forall x \in \mathcal{I}, \quad f(x) = \sum_{k=0}^N \omega_k \mathbf{1}_{\{h_k(x) \geq \rho_k\}}, \quad (3)$$

where N is the number of stumps and $(\omega_k, h_k, \rho_k) \in \mathbb{R} \times \mathcal{F} \times \mathbb{R}$. Here, prior knowledge of the signal is embedded in the choice of the feature set \mathcal{H} . For instance, invariance to changes in illumination may be obtained by using edge detectors while invariance to translation may be achieved by using color or gray-scale histograms estimated over large areas. The resulting strong classifier f is used to classify images of size $W \times H$. In practice, it may also be used for detection, by simply scanning a scene with windows of size $W \times H$.

3.2 Boosting with pose-indexed image features

We consider here a *detection* setup where the scenes for both training and testing consist of images which may contain one or several targets or none at all. Let Θ denote the pose space of the object and let $\theta \in \Theta$ denote a specific pose of that object, encoding all possible parameters including its location in the scene. In this context, an element of a training set takes the form

$$(X^{(i)}, \theta, Y_{\theta}^{(i)}) \in \mathcal{I} \times \Theta \times \{-1, 1\}, \quad (4)$$

where $Y_{\theta}^{(i)}$ is equal to $+1$ if a target is truly visible in $X^{(i)}$ with pose θ , and to -1 otherwise. Ideally such a training set is exhaustive, going through all possible poses $\theta \in \Theta$. Assuming, the only pose parameter of interest is a target's location in a scene, then such a training set enumerates all possible locations of all scenes assigning a positive label where a target is present and a negative one otherwise.

Given a training set as described above, a pose-indexed feature [8] is a function of the form:

$$g_k : \Theta \times \mathcal{I} \rightarrow \mathbb{R}, \quad k = 1, \dots, K. \quad (5)$$

Simply stated, these features depend both on an image and a pose. Next, with a set \mathcal{G} of pose-indexed features, one can construct a boosted pose-indexed classifier of the form

$$\forall \theta \in \Theta, x \in \mathcal{I}, \quad f(\theta, x) = \sum_{k=0}^N \omega_k \mathbf{1}_{\{g_k(\theta, x) \geq \rho_k\}}. \quad (6)$$

Classical object detection, from a single viewpoint, can be formalized in this setting with a two dimensional pose space

$$\Theta = [0, W] \times [0, H]. \quad (7)$$

During training, the features simply translate with the location of every element in the training set and are, in effect, reduced to the features described in 3.1. Detection, at fixed scale, where the scene is parsed at every location, proceeds in a similar manner. Given an image x , detection at a particular threshold T consists of computing a list of alarms

$$\mathbf{A}_T(x) = \left\{ \theta \in \Theta \text{ s.t. } f(\theta, x) \geq T \right\}. \quad (8)$$

This approach extends naturally to arbitrary complex object pose θ while maintaining the joint information between different features. However, it requires the training data to be labelled with the corresponding ground truth, and requires the exploration of pose parameters in test. These drawbacks are further exacerbated by adding more dimensions to the pose space.

4 PROPOSED FRAMEWORK

To retain the benefits of the pose-indexed features without their inherent weaknesses, we treat rigid rotations and deformations, as a collection of hidden variables and simultaneously empower the learning procedure with estimates of those hidden variables. Specifically, we introduce the idea of a pose estimator, which computes a meaningful pose directly from the signal. This computed pose is then used to evaluate various pose-indexed features as is next explained.

4.1 Boosting with pose estimators

We begin by regarding location, which is annotated in training and parsed in testing, in the same way as classical approaches and purely pose-indexed approaches. Let

$$\Theta_1 = [0, W] \times [0, H] \quad (9)$$

represent the aforementioned two-dimensional space standing for the location of the target, and let $\Theta_2 = [-\pi, \pi[$ consist of an orientation in the image plane. Given a pose-indexed feature,

$$g_k : (\Theta_1 \times \Theta_2) \times \mathcal{I} \rightarrow \mathbb{R}, \quad k = 1, \dots, K, \quad (10)$$

a pose estimator is a mapping of the form

$$\eta_m : \Theta_1 \times \mathcal{I} \rightarrow \Theta_2, \quad m = 1, \dots, M. \quad (11)$$

We can now define a pose-indexed image feature γ_{mk} for locations l in the pose space Θ_1 with

$$\forall l \in \Theta_1, x \in \mathcal{I}, \quad \gamma_{mk}(l, x) = g_k((l, \eta_{mk}(l, x)), x). \quad (12)$$

In words, to evaluate a functional γ_{mk} on a scene x for a location $l \in \Theta_1$, we first compute an angle $\theta' = \eta_{mk}(l, x) \in \Theta_2$ and then evaluate g_k for the combined pose (l, θ') and x . These features thus simply have a component which estimates an angle of the target in the image plane. That estimate is then used to evaluate a pose-indexed feature. In practice, different modes of parameterizations are used for the pose-indexed features g_k and each parameterization mode may be seen as effecting a specific type of feature normalization, see Figure 1.

Hence, from a set of cardinality K of pose-indexed features g_k and a set of cardinality M of pose estimators η_m , we create a new set of cardinality MK with features γ_{mk} . This augmented set can then be used with AdaBoost in a straightforward manner. At every iteration, the most successful pose estimator and pose-indexed pair is chosen with the next pair chosen so as to rectify the errors of the previous one resulting in a boosted ensemble of the form

$$\forall l \in \Theta_1, \quad f(l, x) = \sum_{k=0}^N \omega_k \mathbf{1}_{\{g_k((l, \eta_{mk}(l, x)), x) \geq \rho_k\}}. \quad (13)$$

Pose estimator learning and feature learning occurs jointly in a fully integrated fashion: the learning process is allowed to combine several estimates in Θ_2 of an unknown pose and balances different modes of parametrization to reduce classification error. The final detector is highly flexible and able to simultaneously examine the signal in M different ways to determine pose parameters and deform its features accordingly.

4.2 Discussion

Suppose we are tasked with detecting an object class whose pose space may be parameterized by p parameters:

$$\Theta = \Theta_1 \times \dots \times \Theta_p \quad (14)$$

We maintain our definitions for Θ_1 and Θ_2 as the pose spaces of the location of the target and the orientation in the image plane respectively. The additional pose parameters model the rigid rotations and deformations of the target.

Approximating the pose space: By designing a family of pose estimators and allowing the learning method to combine a pose estimator with a pose-indexed feature undergoing a specific type of normalization, the pose space of the object is effectively being approximated with:

$$\Theta \approx \Theta_1 \times \Theta_2^M \quad (15)$$

This is true whether the actual pose space of the object is rich, consisting of deformations and out-of-plane rotations, or very simple consisting say only of in-plane rotations. In the former case of a rich pose space, consisting of say $p - 1$ parameters as described above, the learning method attempts to capture estimates of these parameters using the M pose estimators. In the case of simple in-plane rotations, the M pose estimators all work to capture a single parameter, namely orientation, and are combined and weighted by the learning method.

A deformable detector: It is also worth noting that the final detector that is obtained from our framework spans a very large set of possible configurations. Assuming $M > N$, where we recall that N represents the number of stumps, and allowing for q bins to quantify the response of the pose estimators (see §5.1), the detector possesses a total of

$$q^M \quad (16)$$

instances, each corresponding to a specific instantiation of the M parameter used to deform features. With a basic setup of $q = 8$ and $M = 14$, this results in 4.4×10^{12} different configurations, a very large space which stands in sharp contrast to the single configuration of a rigid model ordinarily constructed by AdaBoost. Whereas one would expect that for a given object class and pose variation, the correlations between the M estimates greatly reduce this space, the same does not hold for the negative class. Thus the entire space of configurations can in fact be utilized by the learning method to discriminate the object class from an arbitrary background.

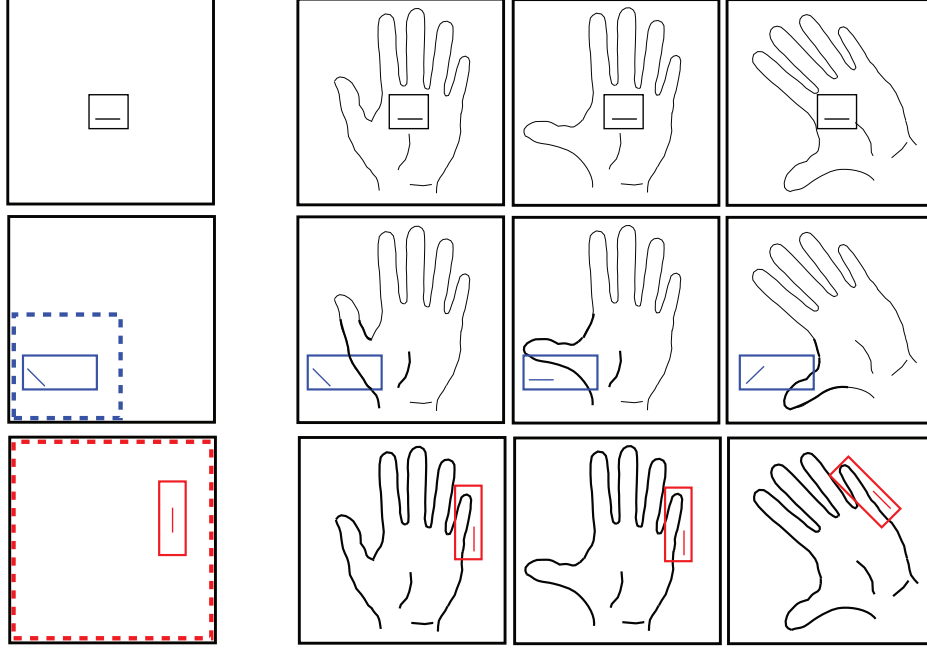


Fig. 1: Our framework mixes three types of edge counting features. Every row shows an example feature from each type along with its extractions for three samples: an open hand, the same hand where the thumb has moved and a rotated version of this case. The example features are shown on the left column: the solid box shows the support of the feature while the solid line within shows the extracted edge orientation. The dashed box shows the area in the image from which the pose estimate is computed, here the dominant edge orientation. This area is also highlighted in every sample by the bolded outline of the hand. **Top row:** a standard feature which checks for the absence of horizontal edges. Note that as the thumb moves and the entire hand is rotated, this features disregards the changes in pose and always checks for the absence of horizontal edges at the same location in the image. **Middle row:** a pose-indexed feature which always has a fixed location but checks for the presence of different edge orientation depending on the dominant edge orientation in the lower-left quadrant of the image. Note how the feature is effectively tracking the thumb. Such features effect so-called “Type I normalization” whereby the extracted edge orientation depends on a pose estimate, see §6.1. **Bottom row:** a pose-indexed feature whose location and edge orientation extraction depend on the dominant edge orientation in the entire image. Note how the feature is effectively tracking the forefinger: it ignores the change in pose as the thumb moves since this has no impact on the global dominant orientation and follows the rotation of the hand in the next sample. Such features effect so-called “Type II normalization” whereby the extracted edge orientation and the feature’s location depend on a pose estimate, see §6.1.

TABLE 1: Various approaches in perspective. **First column:** The predominant strategy which consists of training pose-dedicated classifiers. There, the training data must be fully labelled for the pose θ so that it can be partitioned to train the classifiers, the feature is simply indexed by location and a separate detector is trained for each pose parameter other than location. **Second column:** The pose-indexing framework. There, data must also be annotated while the use of the pose-indexed features allows for training a single classifier indexed by pose on the entire data. Detection must be managed via exhaustive search over the pose parameters. **Third column:** Our framework. Data must only be annotated for location. The combined use of pose-indexed features and pose estimators allows for the training of a single classifier indexed by location. During detection, no search is necessary as the selected pose estimators extract the required pose estimates.

	Predominant Strategy	Pose-indexing	Pose Estimators
Training Data	$(X^{(i)}, \theta, Y_{\theta}^{(i)})$	$(X^{(i)}, \theta, Y_{\theta}^{(i)})$	$(X^{(i)}, l, Y_l^{(i)})$
Feature Set	$h_k : \Theta_1 \times \mathcal{I} \rightarrow \mathbb{R}$	$g_k : \Theta \times \mathcal{I} \rightarrow \mathbb{R}$	$g_k : (\Theta_1 \times \Theta_2) \times \mathcal{I} \rightarrow \mathbb{R}$ $\eta_m : \Theta_1 \times \mathcal{I} \rightarrow \Theta_2$
Training Output	$f_1(l, x), \dots, f_{\ \Theta_1\ }(l, x)$	$f(\theta, x)$	$f(l, x)$
Detection	$\forall l \in \Theta_1, \text{ given } \hat{\theta} \in \Theta,$ $f_{\hat{\theta}}(l, x) = \sum_{k=0}^N \omega_k \mathbf{1}_{\{h_k^{\hat{\theta}}(l, x) \geq \rho_k^{\hat{\theta}}\}}$	$\forall \theta \in \Theta,$ $f(\theta, x) = \sum_{k=0}^N \omega_k \mathbf{1}_{\{g_k(\theta, x) \geq \rho_k\}}$	$\forall l \in \Theta_1,$ $f(l, x) = \sum_{k=0}^N \omega_k \mathbf{1}_{\{g_k((l, \eta_{m_k}(l, x)), x) \geq \rho_k\}}$

Perspective on different approaches: Table 1 puts the various approaches in perspective assuming a general pose space Θ as described in Eqn. 14. Let us consider, by way of example, a target undergoing simple in-plane rotations. The predominant approach in this case is that of Viola and Jones in [31] where 12 rotation specific detectors are trained along with a pose estimator returning an estimate of the target’s orientation in the image plane. The pose-indexed approach in this case would train a single detector with features that rotate according to the labelled pose. In testing, one would simply test all possible rotations at all possible locations and retain the maximum response. In contrast, our approach would initiate training on the unlabeled training data and M pose parameters are used to approximate the target’s rotation in the image plane: each pose-indexed feature would obtain its pose information from one of the M parameters. Those same parameters are extracted during testing, and used to evaluate their associated pose-indexed features.

Our implementation, as described in §5, should not be understood as dealing with the full range of out-of-plane rotation: for example, one should not apply our implementation to build a single, monolithic, deformable detector capable of simultaneously detecting a front view car and a side view car. As mentioned in §2, in such a setting, the view-based approaches are a sensible design strategy. The later strategy should be combined with our proposed deformable detector to reduce data fragmentation and thereby improve detection performance. We note that the method in [6] in fact mixes a view-based approach with deformable parts. However, a very limited number of parts are used and much as the purely pose-indexed approaches, it requires the exploration of possible configurations in testing. The method is additionally designed to leverage higher resolution content. In comparison, our method uses hundreds of deformable features, does not require exploration of pose parameters in testing and is capable of providing reliable detection even in low resolution.

5 IMPLEMENTATION DETAILS

The specifics of our implementation are given in this section. We follow the same notation as that of previous sections.

5.1 Standard Feature Set

We describe here two types of standard image features, not yet indexed by a pose. A scene x is preprocessed by computing and thresholding the derivatives of the image intensity to obtain an edge image. The orientation of these edges are further quantized into q bins, resulting in q edge maps. Let ϕ denote the possible orientations of an edge on $\Phi = [-\pi, \pi]$, and let $\hat{\Phi} = \{0, \frac{2\pi}{q}, \frac{4\pi}{q}, \dots, (q-1) * \frac{2\pi}{q}\}$ denote the possible orientations of a *quantized* edge.

Now $\forall e \in \hat{\Phi}, x \in \mathcal{I}, l \in \{1, \dots, W\} \times \{1, \dots, H\}$, let

$$\xi_e(x, l) \in \{0, 1\}, \quad (17)$$

denote the presence of an edge with quantized orientation e at pixel l in image x . We assume $\xi_e(x, l)$ is equal to 0 if the location l is not in the image plane. Thus, each $\xi_e(x, l)$ is simply a map of edges with quantized orientation e , see

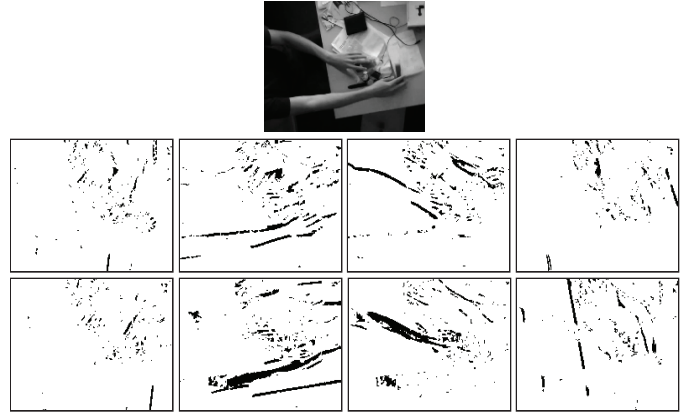


Fig. 2: From the original gray-scale image (top), we compute eight edge maps (two lower rows), corresponding to eight different orientations of a simple edge detector. Integral images of these edge maps are used to efficiently compute proportions of edges in rectangular windows.

Fig. 2 for $q = 8$. We also consider a smoother version of $\xi_e(x, l)$ defined as:

$$\bar{\xi}_e(x, l) = \max(0, \cos(\phi - e)) \quad (18)$$

In this case, each edge with orientation ϕ at pixel l in image x contributes a soft value to each edge map. We again assume $\bar{\xi}_e(x, l)$ to be equal to 0 if the location l is not in the image plane. In practice, the hard edge map based feature perform poorly with high q . This becomes immediately obvious when we consider that with a fine discretization, edge orientations become increasingly noisy. Soft-features, which allow for soft votes for every edge, become useful with high q . For the remainder of this paper, the discussion is presented with respect to the hard edge maps though all equations extend equally to the soft edge maps by simply substituting $\xi_e(x, l)$ with $\bar{\xi}_e(x, l)$.

Our features, similar to those of [1], compute the ratio of edges of a particular orientation within a sub-window of the detector’s $r \times r$ square of interest, with respect to the total number of edges within the same sub-window. Let R denote such a sub-window of random size and location contained in $\{1, \dots, r\} \times \{1, \dots, r\}$ plane. Our features are entirely parameterized by the sub-window R and the edge type e and are defined as:

$$h_{R,e}(x) = \sum_{m \in R} \xi_e(x, m) / \sum_{d \in \hat{\Phi}, m \in R} \xi_d(x, m). \quad (19)$$

These features give the classifier the ability to check for the presence of outlines and textures and can be computed in constant time using q integral images, one for each edge map.

5.2 Pose-Indexed Image Features

From the image features described above, we define a set of features indexed by a location in the image plane and an orientation. We define $\Theta_1 = \{1, \dots, W\} \times \{1, \dots, H\}$ and

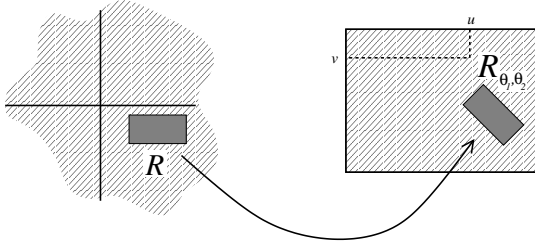


Fig. 3: From a rectangular window R and a pose (u, v, θ_2) , we define an indexed window R_{l, θ_2} . Here $\theta_2 = \pi/4$.

$\Theta_2 = [-\pi, \pi[$. Given a rectangular sub-window R , and poses $l = (u, v) \in \Theta_1$, and $\theta_2 \in \Theta_2$, we define

$$R_{l, \theta_2} \quad (20)$$

as the rectangular window in the image plane obtained by applying a rotation of angle θ_2 and a translation (u, v) .

Similarly, given an edge orientation $e \in \hat{\Phi}$ and an angle $\theta_2 \in \Theta_2$, we define

$$e_{\theta_2} \quad (21)$$

as the orientation obtained after a rotation of θ_2 is applied to the edge, that is the edge orientation in $\hat{\Phi}$ closest to $e + \theta_2$.

With the above notation, we can define a set of pose-indexed features from $h_{R, e}$ introduced above, with

$$g_{R, e}((l, \theta_2), x) = h_{R_{l, \theta_2}, e_{\theta_2}}, \quad (22)$$

which is, the proportion of edges with a rotated edge orientation in the translated and rotated rectangular window.

We note that the orientation of the resulting window is again quantified with a resolution of q for computational reasons. Rotations of angles proportional to $\pi/2$ and $\pi/4$ are ideal, see Fig. 3. For other angles, rotations are approximated for maximum overlap with the ideal case. The features themselves can be computed in constant time with $2q$ integral images: q integral images for each edge map and an additional q for each edge map rotated by $\pi/4$.

5.3 Pose Estimators

We define a family of pose estimators which estimate a meaningful orientation $\theta_2 \in \Theta_2$ from a location $l = (u, v)$. Our pose estimators compute the dominant edge orientation in a particular window Λ contained in the neighborhood of l . More precisely, we define

$$\eta_{\Lambda}(l) = \arg \max_{e \in \hat{\Phi}} h_{\Lambda, e}, \quad (23)$$

which computes the dominant edge orientation θ_2 in the window Λ translated according to l . Given the $\{1, \dots, r\} \times \{1, \dots, r\}$ plane \mathbf{r} , we define 14 regions for the pose-estimators corresponding to the complete square, the four regular sub-squares, and the nine regular sub-squares, which leads to 14 different pose-estimators, as shown in Fig. 4. Note that the estimated pose is quantified with the same number of bins q so as to allow for the reuse of the integral images.

In addition to these 14 pose estimators, we defined 3 more global pose estimators for our experiments with the face data

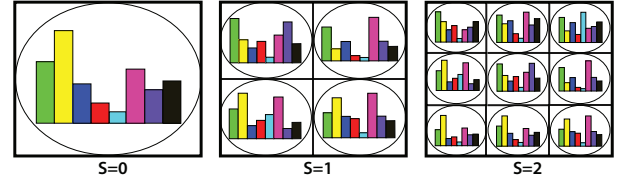


Fig. 4: Our family of pose estimators. Given the square of interest of size $r \times r$ centered on l , there are 14 pose estimators in total operating: each one computes the dominant edge orientation θ_2 in one of the sub-squares at three different scales S .

sets. They determine the global orientation in the image plane by looking for the axis which maximizes symmetry of the two-half images using various metrics.

6 EXPERIMENTS

To evaluate the performance of our proposed learning strategy, experiments were performed on three different data sets: video sequences of hands, aerial images of cars and face images. For all data sets, we compare the performance of our method against that of a standard boosting procedure with access to the *same* ground truth. In the case of the aerial images of cars, where pose variation consists mainly of pure in-plane rotations, we also compared the performance of our method with the optimal pose normalization scheme: a try-all-rotations detector trained on manually aligned data. In what follows, the specifics of our experimental setup are given and the results of our experiments provided.

6.1 Learning

The standard AdaBoost learning procedure is used. Two boolean flags are added to the definition of our augmented pose-indexed features. The first indicates if the feature is to take the pose estimate into account. If so, the second flag specifies if the feature's window is to be registered according to the rotation described in § 3.2. Given a pose, $(l, \theta_2) \in (\{1, \dots, W\} \times \{1, \dots, H\}) \times [-\pi, \pi[$, three types of features are hence obtained:

- The first ignores the pose estimate and thus reduces to the standard feature as it simply translates its window with l .
- The second considers the pose estimate θ_2 insofar as its edge orientation type is concerned while still translating its window with l .
- The third translates its window with l , applies a rotation to the latter and changes its edge orientation type according to θ_2 .

We refer to the second and third items as Type I normalization and Type II normalization respectively.

The selection of the stump at every iteration of AdaBoost results from examining 1000 of these features. The threshold ρ_i of the selected stumps is optimized through an exhaustive search. The boolean flags are naturally selected randomly, with probability 0.5. The pose estimator is also chosen randomly: the scale at which it examines the signal is first chosen

uniformly and the same is true for the sub-square over-which orientation is computed (among 1, 4 or 9 possible), see Fig. 4. Finally, the window R and the edge orientation e are also chosen uniformly at random. In all our experiments, a single AdaBoost stage is trained with the bootstrapping procedure described in [8]: this allows us to avoid the difficulties associated with training and tuning a cascade. All of the results are averaged over five independent runs. Since we observed the absence of over-fitting, we did not optimize learning parameters through cross-validation. Learning and testing algorithmic outlines are given in Algorithms 1 and 2 below.

Algorithm 1 Learning Outline

Given training data $(X^{(i)}, l, Y_l^{(i)})$, a set of K pose-indexed features g_k and a set of M pose-estimators η_m .

- 1: Initliaze weights $\alpha_{1,i,l} = \frac{1}{2a}, \frac{1}{2b}$ where a and b are the total number of positive and negative examples respectively.
- 2: **for** $k = 1$ **to** N **do**
- 3: **for** $t = 1$ **to** 1000 **do**
- 4: Choose at random a pose-indexed feature $g_k^{(t)}$, a pose estimator $\eta_m^{(t)}$ and a normalization mode. Evaluate weighted classification error after threshold optimization $\rho_k^{(t)}$:

$$\epsilon_t = \sum_{i,l} \alpha_j \left| \mathbf{1}_{\{g_k^{(t)}((l, \eta_m^{(t)}(l, x^{(i)})), x^{(i)}) \geq \rho_k^{(t)}\}} - y_l^{(i)} \right|$$

- 5: **end for**
- 6: Define g_k, η_{m_k}, ρ_k as the minimizers of ϵ_t .
- 7: Update data weights:

$$\alpha_{k+1,i,l} = \alpha_{k,i,l} \beta_t^{1-e_{i,l}}$$

where $e_{i,l} = 0$ if image $x^{(i)}$ at location l is classified correctly and $e_{i,l} = 1$ otherwise. $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.

- 8: Set $\omega_k \leftarrow \frac{1}{2} \log \frac{1}{\beta_t}$
- 9: Normalize data weights $\alpha_{k+1,i,l} \leftarrow \frac{\alpha_{k+1,i,l}}{\sum_j \alpha_{k+1,i,l}}$
- 10: **end for**
- 11: The final detector is given by:

$$f(l, x) = \sum_{k=0}^N \omega_k \mathbf{1}_{\{g_k((l, \eta_{m_k}(l, x)), x) \geq \rho_k\}}$$

Algorithm 2 Detection Outline

Given a patch from image x and location l .

- 1: Evaluate all M pose estimators η_m .
- 2: Evaluate strong classifier:

$$f(l, x) = \sum_{k=0}^N \omega_k \mathbf{1}_{\{g_k((l, \eta_{m_k}(l, x)), x) \geq \rho_k\}}$$

6.2 Error rates

Error rates were computed in a conservative fashion. A detection is a true alarm if its location is within a certain distance from the target and a false alarms otherwise. The considered distance is half the length of the detector's square window of interest. In several frames, in both the hand and the car data sets, two targets may lie within the above mentioned distance. In this scenario, if only one alarm is raised, a miss is counted. In all our experiments we have chosen to use false alarm rate for uniformity instead of the number of false alarms. Given that all images were scanned by steps of two pixels, a conversion to this second measure can be obtained simply by multiplying the rate with the size of the image, given below for each data set, and dividing by four.

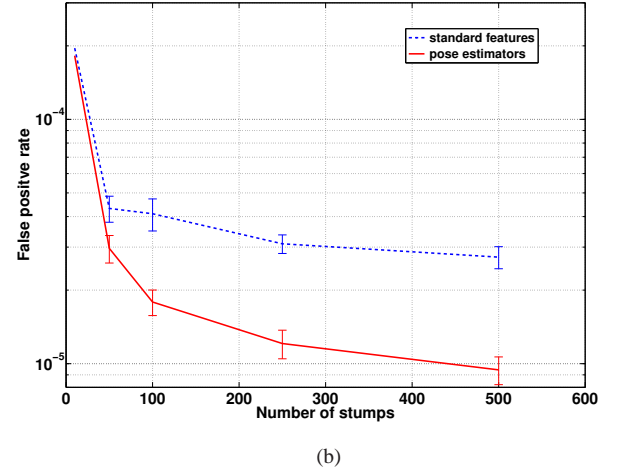
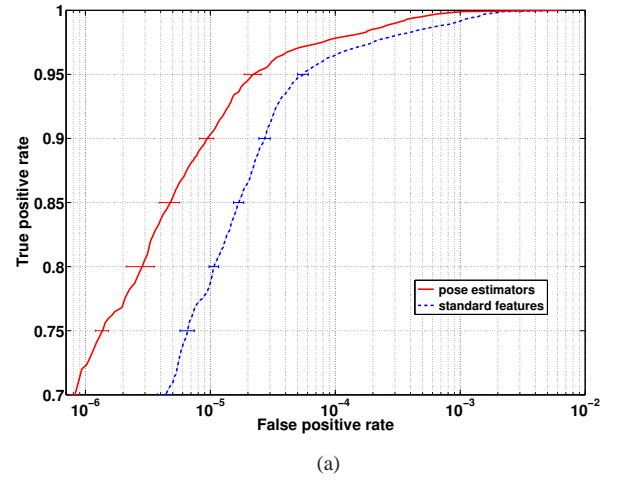


Fig. 5: Performance of our learning framework compared with a standard boosting framework for hardware-specialized camera (training and testing). Figure (a) displays true-positive rate as a function of the false alarms rate on a log scale. Figure (b) displays the false alarm rate at 90% true positive rate as a function of the number of stumps. In both figures, the thin blue curve corresponds to the performance of the standard feature set while the thick red curve shows the performance of the detector using the combination of pose-indexed features and pose-estimators.

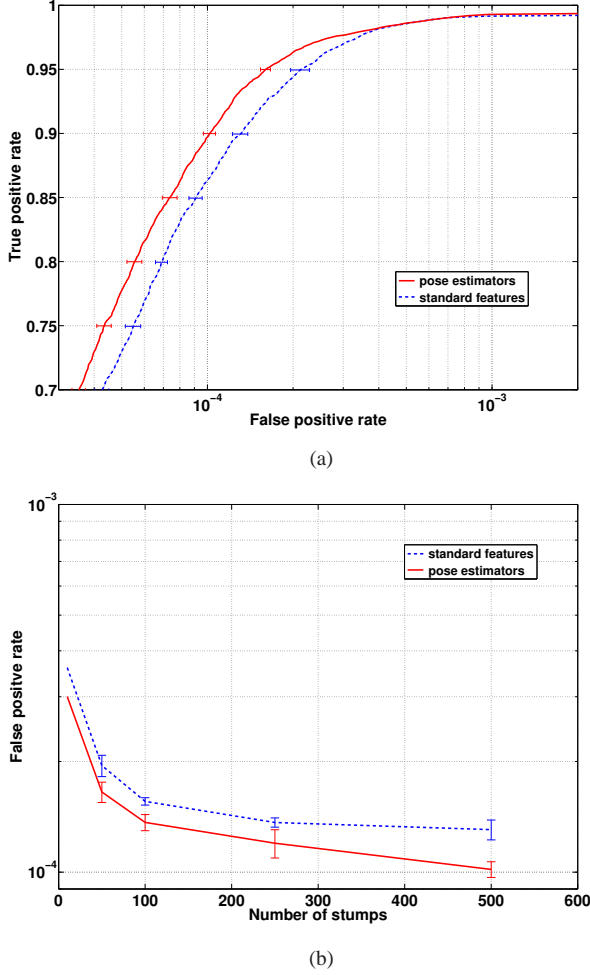


Fig. 6: Performance of our learning framework compared with a standard boosting framework for the webcam data set (training and testing). Figure (a) displays true-positive rate as a function of the false alarms rate on a log scale. Figure (b) displays the false alarm rate at 90% true positive rate as a function of the number of stumps. In both figures, the thin (dashed) blue curve corresponds to the performance of the standard feature set while the thick red curve shows the performance of the detector using the combination of pose-indexed features and pose-estimators.

6.3 Experiments on Hand Video Sequences

6.3.1 Data

We carried out our tests on two data sets. Each data set contains two hand sequences: one sequence is used for training, the other is used for testing. Our first data set was obtained from a hardware-specialized camera [22] which directly computes edges and is to an extent illumination invariant. These sequences have a resolution of 128×160 , a frame rate of approximately 7 *fps* and a duration of 4 minutes. The scene consists of a piece of heavy machinery with a few moving parts and clutter. Our second data set was obtained from a standard webcam. These sequences have a resolution of 144×192 , a frame rate of approximately 10 *fps* and a duration of 5 minutes. The scene consists of a typical disorderly office desk.

6.3.2 Setup

The boosting stage is trained with 1500 positive examples and 150,000 negative examples for the hardware-specialized data set. Similarly, the boosting stage corresponding to the webcam data set is trained with 1800 positive samples and 180,000 negative samples. Learning was carried out up to 500 stumps for both data sets. Hard Edge maps were used with $q = 8$.

6.3.3 Results

We compared the performance of our augmented feature set with that of the standard features. As shown in Figures 5 and 6, incorporating pose estimator learning with feature learning provides with a significant gain in false positive rate at *all* detection rates and for both data sets. Indeed our method is able to capture the strong changes in appearance of the hand where the standard features fail. Most notably, at 90% true positive rate our first hardware-specialized data set, our method raises 9.4×10^{-6} false alarms per frame versus 2.7×10^{-5} for the standard features, a gain of approximately 180%. Some example frames, chosen uniformly at random, are shown in Figure 17 and 18 for both data sets.

Figures 7 and 8 show some statistics with respect to the type of features selected by AdaBoost. We note that the percentage of features operating with pose estimators quickly rises with each boosting step and stabilizes at approximately 70% leaving 30% for the standard features. This is intuitively meaningful: with each boosting step, harder samples remain to be classified and more of the augmented features are brought into play. We also note that the pose estimators are utilized relatively uniformly and across both normalization schemes. The most frequently selected features utilize the large scale pose estimator with type II normalization: these features account for the in-plane rotation that is present throughout the sequence. The remaining augmented features, utilized at over 80%, account for the deformations of the hands.

Table 2 shows run-times obtained for the proposed framework on the webcam test sequence with varying number of stumps. We note that the code was not optimized for best performance and that implementing a simple early-rejection cascade, for example, would result in significant speed-ups while maintaining performance constant.

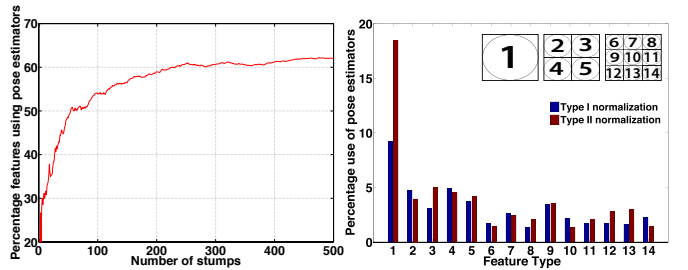


Fig. 7: Frequency of features selected by AdaBoost during training for the hardware-specialized camera. Left: the rate at which features operating with pose estimators are selected as a function of the number of stumps. Right: the allotment of features to each pose estimator.



Fig. 9: Some examples of cars from our test data set taken uniformly at random across the entire set

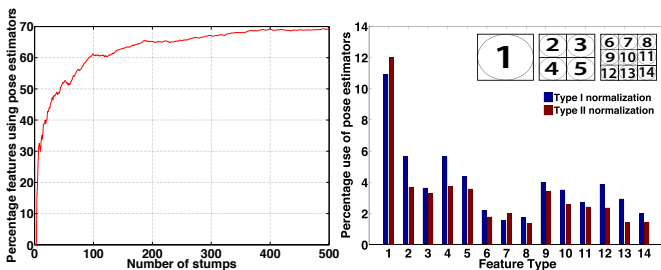


Fig. 8: Frequency of features selected by AdaBoost for the webcam data set. Left: the rate at which features operating with pose estimators are selected as a function of the number of stumps. Right: the allotment of features to each pose estimator.

TABLE 2: Run-times for the webcam dataset on a general-purpose Intel[®] Xeon[®] L5420 processor, 2.50Ghz.

Number of stumps	Frame processing time (ms)	FPS
100	28.38	35
200	49.07	20
300	69.55	14
400	88.32	11
500	107.56	9

6.4 Experiments on Aerial Images of Cars

6.4.1 Data

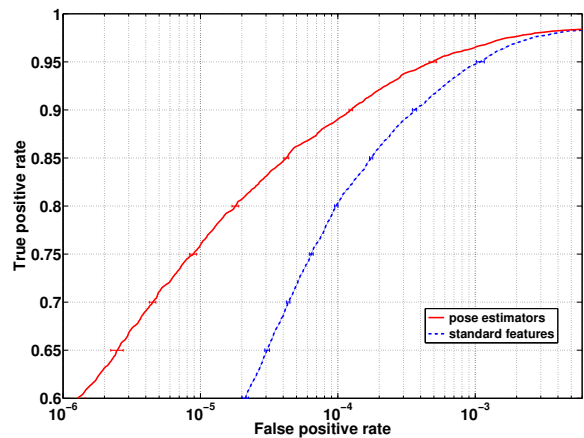
Our data set consists of 100 aerial images of resolution 1064×744 collected over Lausanne and Geneva at a constant altitude. The images contain approximately 3000 cars, parked or in motion, in a highly challenging urban environment: shadows are cast by buildings and greenery often occlude over half the targets. In addition, cars are customarily parked side-by-side leaving very little space in between rendering detection even more troublesome. Some sample patches taken uniformly at random are shown in figure 9. The pose variation we are interested in here is in-plane rotation as cars can be found in any orientation.

6.4.2 Setup

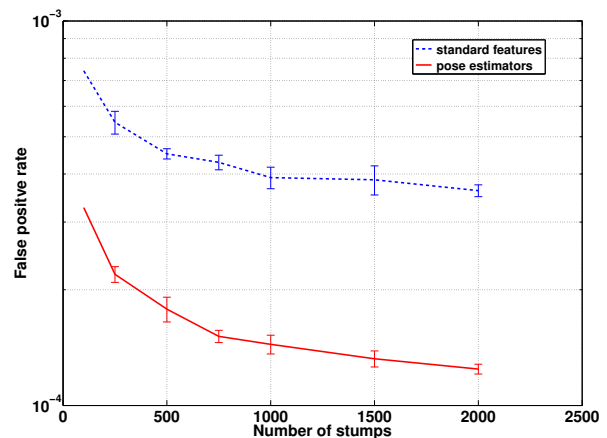
Images over Lausanne were used for training while images over Geneva were used for testing. The boosting stage is trained with 1500 positive examples and 2,000,000 negative examples while learning was carried up to 2000 stumps. Hard Edge maps were used with $q = 16$.

6.4.3 Results

We compared the performance of our augmented feature set with that of a standard boosting procedure with access to



(a)



(b)

Fig. 10: Performance of our learning framework compared with a standard boosting framework for the car data set. Figure (a) displays true-positive rate as a function of the false alarms rate on a log scale. Figure (b) displays the false alarm rate at 90% true positive rate as a function of the number of stumps. In both figures, the thin blue curve corresponds to the performance of the standard feature set while the thick red curve shows the performance of the detector using the combination of pose-indexed features and pose-estimators.

the same ground truth. As shown in figure 10, incorporating pose estimator learning with feature learning provides with a significant gain in false positive rate at *all* detection rates and for both data sets. Indeed, at 90% true positive rate, our method raises 1.2×10^{-4} false alarms per frame versus 3.6×10^{-4} for the standard features, a gain of approximately 200%. Gains of an order of magnitude are observed at true positive rates below 70%. Some example detections, are shown in Figure 19.

Figure 11 shows some statistics with respect to the type of features selected by the AdaBoost learning procedure. We note that the percentage of features operating with pose estimators starts off at a 100% and stabilizes at approximately 75%. This behavior is rather different from the one observed for the hand video sequences and can be explained by the fact that features using the global pose estimator with type II normalization perform in-plane rotation normalization. Given that the pose variations in the car images consist mainly of in-plane rotations, the only features that offer AdaBoost good error rates are immediately that variety. This was empirically confirmed as the first 10 features selected by AdaBoost are consistently effecting type II normalization with our global pose estimator. We note however that as more and more stumps are added, the pose estimators are utilized relatively uniformly and across both normalization schemes. Even though, we are only faced with in-plane rotations, none of the pose estimators are accurate individually: AdaBoost hence combines the various pose estimators in order to compensate for this deficiency.

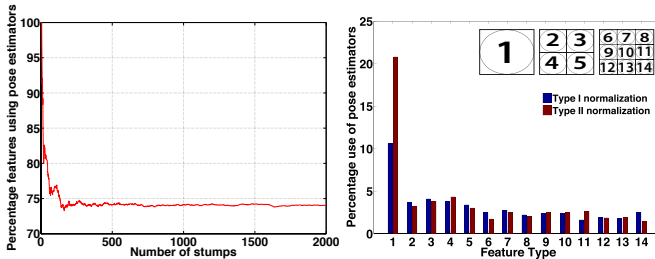


Fig. 11: Frequency of features selected by AdaBoost for the car data set. Left: the rate at which features operating with pose estimators are selected as a function of the number of stumps. Right: the allotment of features to each pose estimator.

We also compared our detector with an optimal pose normalization scheme. This consists in manually aligning the car patches for training and parsing all possible rotations of the images in testing. Note that for both training and testing the images were rotated and bilinear interpolation was performed so as to avoid possible artifacts. As can be seen in figure 12, our framework outperforms the try-all-rotations detector at high true positive and low true positive. The try-all-rotations detector performs better in the true positive range 0.75 – 0.95. Note however that as shown [31], a try-all-rotations detector trained on aligned data exhibits slightly higher accuracy than the state of the art two stage approach: the advantage of the latter is that a search over all rotations is not necessary. Both methods require pose annotation of the data for training, which our framework forgoes. We also note that our framework performs as well as the purely-pose-

indexed approach: there, pose-indexed features are used to perform in-plane normalization in training, according to the labelled pose, and all rotations are explored in testing.

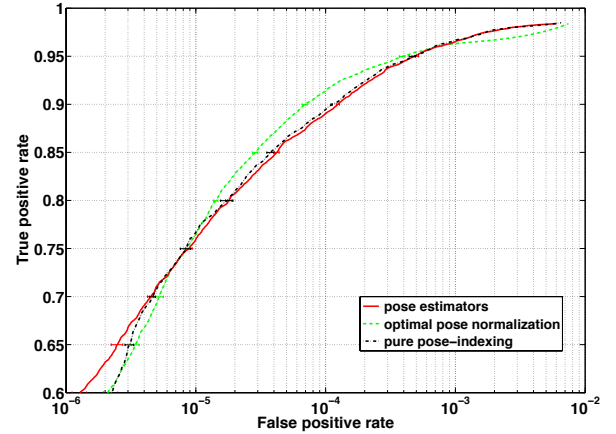


Fig. 12: Performance of our learning framework compared with the optimal pose normalization scheme and a purely pose-indexed approach for the car data set. The figure displays true-positive rate as a function of the false alarms rate on a log scale. The thin green curve corresponds to the performance of the standard feature set, the thick red curve shows the performance of the detector using the combination of pose-indexed features and pose-estimators and the thin black curve shows the performance of a purely pose-indexed approach.

6.5 Experiments on face images

6.5.1 Data

Our data here consists of 8000 face images of size 48×48 . These faces differ from the standard data sets in that the images contain most of the head, from forehead including the chin and jaw lines, as well as some background. More importantly, the images were collected so as to include generally upright and generally frontal faces but without paying much attention to the pose. Thus the data set contains some variation in terms of in-plane rotation as well as out-of-plane rotation. Some examples are shown in 13. We believe that such a data set captures well the inherent natural variations that exist in photographs.

6.5.2 Setup

For this data set, we ran classification experiments as opposed to detection. Thus for training, we used 4000 positive samples and 6000 negative samples. For testing 4000 positive samples were tested with approximately 6,000,000 negative samples to ensure stable and meaningful false positive rates. Negative data was collected randomly from large images that do not contain faces. Learning was carried up to 1500 stumps and experiments were performed using soft edge maps with $q = 72$, corresponding to a quantization of edge orientation of 5 degrees. Such a fine discretization was required in order to capture the rigid rotations variations which exhibit a very small standard deviation of approximately 10 degrees.



Fig. 13: Some examples of faces from our test data set taken uniformly at random across the entire set

6.5.3 Results

We compared the performance of our augmented feature set with that of a standard boosting procedure with access to the same ground truth. As can be seen in figure 14, our method performs as well as a standard boosting framework for true positive rates above 87.5%. Below that true positive rate however, our framework provides very significant gains. Indeed, at 82.5% true positive rate, our method raises 7.7×10^{-7} false alarms per frame versus 1.6×10^{-6} for the standard features, a gain of approximately 100%. Note also that at approximately 81% true positive, our framework raises 0 alarms and all the 6,000,000 negative patches are correctly classified. The same behavior is noted for the standard boosting framework, though at 72.5% true positive. We note that pose estimators were again utilized relatively uniformly and constituted 40% of all features selected.

6.6 Assessing the effects of Joint Learning

We are interested in analyzing the benefits brought about by the joint pose estimator and feature learning we propose. In particular, we are interested in the performance that would result from constraining the pose-indexed features to utilize only one pose estimator and perform only one type of normalization. To this end, we considered the case where features are forced to employ the global pose estimator and perform type II normalization. This is essentially a scheme that attempts to perform in-plane normalization based on a pose estimate obtained from a hand-crafted rule.

This setup allows us to truly understand where the gains in performance originate from. For fairness, these experiments were performed on two of our data sets: the car data set where most of the pose variation is in-plane rotation and the hand webcam data set where pose variation consists mainly of deformations though in-plane rotation is present throughout the sequence, given that there are two hands with different orientations.

Figure 15 shows the results obtained for the car data set. As expected the constrained scheme performs between our framework and the standard boosting framework, though closer to the former. Upon first examination, this was surprising since we observed that the global pose estimator is prone to error even though in pose is visible in most samples. Upon closer examination, we noted that the errors of the global pose estimator are consistent in that the latter fails on clusters of samples exhibiting the same difficulties, namely occlusion or strong shading. The AdaBoost learning procedure readily

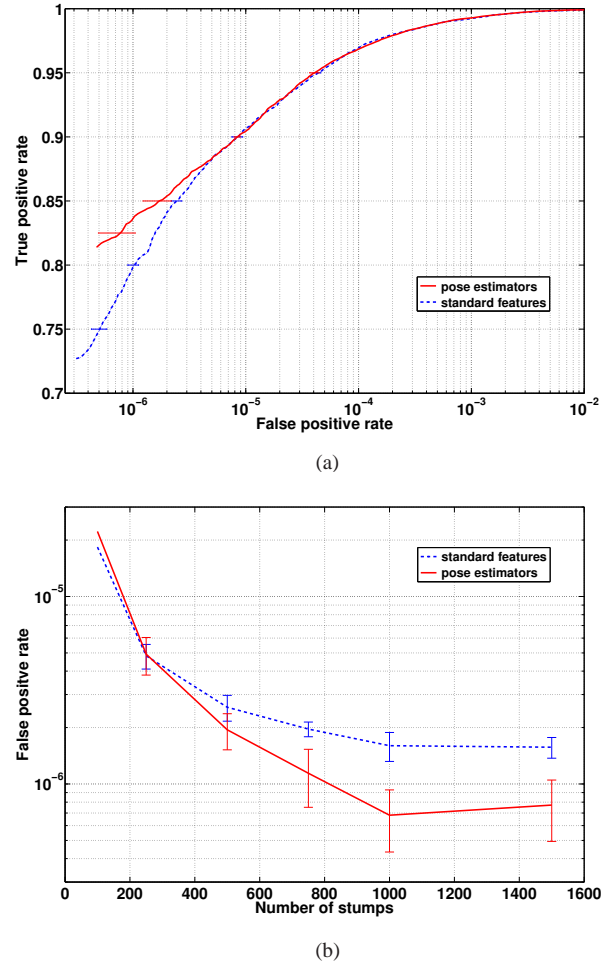


Fig. 14: Performance of our learning framework compared with a standard boosting framework for our face data set. Figure (a) displays true-positive rate as a function of the false alarms rate on a log scale. Figure (b) displays the false alarm rate at 82.5% true positive rate as a function of the number of stumps. In both figures, the thin blue curve corresponds to the performance of the standard feature set while the thick red curve shows the performance of the detector using the combination of pose-indexed features and pose-estimators. Soft edge map features are used.

cope with this situation by placing features at consistent locations for each cluster and weighing them appropriately. The previous observation notwithstanding, it is clear that

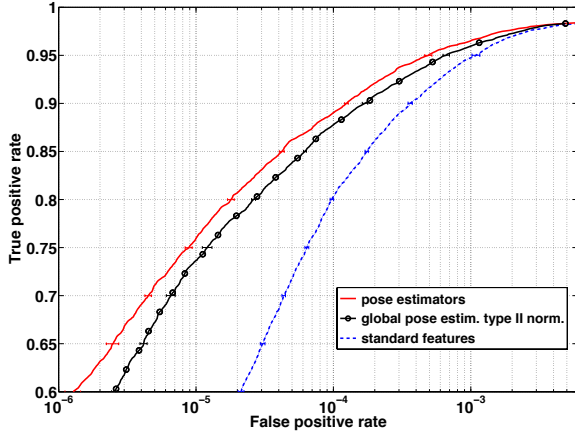


Fig. 15: Performance on the car data set of our learning framework compared with a scheme utilizing the same framework but where features are constrained to employ the global pose estimator and effect type II normalization for the car data. The figure displays true-positive rate as a function of the false alarms rate on a log scale. The thick red curve shows the performance of the detector using the combination of pose-indexed features and pose-estimators and the black curve, with circular markers, shows the performance of the global pose estimator only scheme. The thin blue curve corresponds to the performance of the standard feature set.

the joint learning we propose brings significant benefits in this case with gains of 100% compared to a hand-designed normalization rule.

Figure 16 shows the results obtained for the hand webcam data. As can be seen, the performance of the constrained scheme is far worse than that of our framework. Surprisingly, it is even worse than the performance of the standard boosting framework. This can be explained by the fact that the global pose estimator is unreliable, returning nearly random poses, for a large number of samples. Thus, the global pose estimator is unable to account on its own for the in-plane rotation variations that are present in the data. This is in addition to the fact that constraining features to use the global pose estimator and effect type II normalization offers no possibility to handle deformation.

7 CONCLUSION

We introduced a novel object-detection strategy to handle complex changes in target pose. Our method consists of designing a series of pose estimators able to directly compute an orientation in the image plane, and to allow the learning process to chose the most efficient combinations of pose estimators and pose-indexed features. This procedure produces a detector able to modulate its features according to the image signal hence adapting to variations in appearance and local deformations without the need for fragmenting the data during training, nor visiting additional pose parameters during detection.

A simple class of features truly invariant to rotation would compute the maximum proportion of edges over all possible

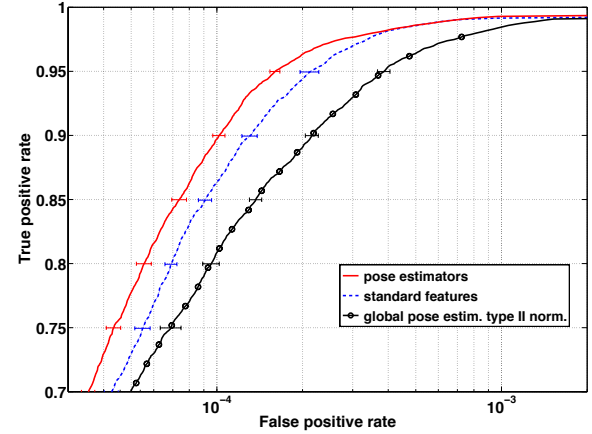


Fig. 16: Performance of our learning framework compared with a scheme utilizing the same framework but where features are constrained to employ the global pose estimator and effect type II normalization for the hand (webcam) data. The figure displays true-positive rate as a function of the false alarms rate on a log scale. The thick red curve shows the performance of the detector using the combination of pose-indexed features and pose-estimators and the black curve, with circular markers, shows the performance of the global pose estimator only scheme. The thin blue curve corresponds to the performance of the standard feature set and is shown for reference

orientations in a fixed sub-window. The pose estimators we use, as defined in §5.3, provide the same operator when the windows of the pose-estimator and the pose-indexed features are identical. Hence, the features we have designed form a super-set of simple truly invariant features, as they are able to estimate the orientation in a window, and evaluate the response for that orientation in another one. Extension of this work can follow two different axes. The first is to consider the use of more complex pose-estimators, going beyond the direct use of the edge counting features. The second axis will consist of investigating the relationship between standard invariant features and alternatives of the combination of pose-indexed features and pose-estimator we propose here, as stated above. By deconstructing standard image invariants in the same way, we may exhibit new valuable classes of both pose-indexed features and pose estimators.

REFERENCES

- [1] Y. Amit, D. Geman, and B. Jedynek. Efficient Focusing and Face Detection. *Face Recognition: From Theory to Applications*, 1998.
- [2] P. Carbonetto, G. Dorkó, C. Schmid, H. Kück, and N. Freitas. Learning to recognize objects with little supervision. *International Journal of Computer Vision*, 77(1-3):219–237, 2008.
- [3] O. Chum and A. Zisserman. An Exemplar Model for Learning Object Classes. In *Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.
- [4] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Conference on Computer Vision and Pattern Recognition*, 2005.
- [5] Gy. Dorkó and C. Schmid. Selection of scale-invariant parts for object class recognition. In *International Conference on Computer Vision*, page 634, 2003.

- [6] P. Felzenszwalb, D. Mcallester, and D. Ramanan. A Discriminatively Trained, Multiscale, Deformable Part Model. In *Conference on Computer Vision and Pattern Recognition*, June 2008.
- [7] R. Fergus, P. Perona, and A. Zisserman. Object Class Recognition by Unsupervised Scale-Invariant Learning. In *Conference on Computer Vision and Pattern Recognition*, pages 264–271, July 2003.
- [8] F. Fleuret and D. Geman. Stationary Features and Cat Detection. *Journal of Machine Learning Research*, 9:2549–2578, 2008.
- [9] Y. Freund and R.E. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. In *European Conference on Computational Learning Theory*, pages 23–37, 1995.
- [10] M. Kolsch and M. Turk. Analysis of Rotational Robustness of Hand Detection With a Viola-Jones Detector. *Journal of Machine Learning Research*, 3:107–1103, August 2004.
- [11] A. Kushal, C. Schmid, and J. Ponce. Flexible object models for category-level 3d object recognition. *Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.
- [12] B. Leibe and B. Schiele. Scale-Invariant Object Categorization Using a Scale-Adaptive Mean-Shift Search. In *German Association for Pattern Recognition*, 2004.
- [13] S. Z. Li and Z. Zhang. Floatboost learning and statistical face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26, 2004.
- [14] S. Z. Li, L. Zhu, Z. Zhang, A. Blake, H.Z. Zhang, and H. Shum. Statistical Learning of Multi-View Face Detection. *European Conference on Computer Vision*, pages 67–81, 2002.
- [15] J. Liebelt, C. Schmid, and K. Schertler. Viewpoint-independent object class detection using 3d feature maps. In *Conference on Computer Vision and Pattern Recognition*, 2008.
- [16] A. Mohan, C. Papageorgiou, and T. Poggio. Example-Based Object Detection in Images by Components. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(4):349–361, 2001.
- [17] A. Opelt, M. Fussenegger, A. Pinz, and P. Auer. Weak hypotheses and boosting for generic object detection and recognition. In *European Conference on Computer Vision*, pages 71–84, 2004.
- [18] M. Ozuysal, V. Lepetit, and P. Fua. Pose Estimation for Category Specific Multiview Object Localization. In *Conference on Computer Vision and Pattern Recognition*, June 2009.
- [19] C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.
- [20] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *Conference on Computer Vision and Pattern Recognition*, 1996.
- [21] H. A. Rowley, S. Baluja, and T. Kanade. Rotation Invariant Neural Network-Based Face Detection. *Journal of Machine Learning Research*, page 963, 1998.
- [22] P.-F. Ruedi, P. Heim, F. Kaess, E. Grenet, F. Heitger, P.-Y. Burgi, S. Gyger, and P. Nussbaum. A 128 X 128 Pixel 120-Db Dynamic-Range Vision-Sensor Chip for Image Contrast and Orientation Extraction. *Solid-State Circuits, IEEE Journal of*, 38(12):2325–2333, December 2003.
- [23] S. Savarese and L. Fei-Fei. 3D Generic Object Categorization, Localization and Pose Estimation. In *International Conference on Computer Vision*, 2007.
- [24] H. Schneiderman and T. Kanade. A Statistical Method for 3D Object Detection Applied to Faces and Cars. *Journal of Machine Learning Research*, 1:746–7511, 2000.
- [25] H. Schneiderman and T. Kanade. Object Detection Using the Statistics of Parts. *Computer Vision and Image Understanding*, 2002.
- [26] B. Stenger, A. Thayananthan, P.H.S. Torr, and R. Cipolla. Filtering Using a Tree-Based Estimator. In *International Conference on Computer Vision*, pages 1063–1070, 2003.
- [27] B. Stenger, A. Thayananthan, P.H.S. Torr, and R. Cipolla. Estimating 3D Hand Pose Using Hierarchical Multi-Label Classification. *Image Vision Comput.*, 25(12):1885–1894, 2007.
- [28] H. Su, M. Sun, L. Fei-Fei, and S. Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *International Conference on Computer Vision*, 2009.
- [29] A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. Van Gool. Towards Multi-View Object Class Detection. In *Conference on Computer Vision and Pattern Recognition*, 2006.
- [30] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: Efficient boosting procedures for multiclass object detection. In *Conference on Computer Vision and Pattern Recognition*, 2004.
- [31] P. Viola and M. Jones. Fast Multi-View Face Detection. Technical report, MERL, 2003.
- [32] P. Viola and M. Jones. Robust Real-Time Face Detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.
- [33] P. Viola, M. Jones, and D. Snow. Detecting Pedestrians Using Patterns of Motion and Appearance. In *International Conference on Computer Vision*, pages 734–741, 2003.
- [34] P. Yan, S. M. Khan, and M. Shah. 3d model based object class detection in an arbitrary view. In *International Conference on Computer Vision*, 2007.
- [35] J. Zhang, S. K. Zhou, L. McMillan, and D. Comaniciu. Joint Real-Time Object Detection and Pose Estimation Using Probabilistic Boosting Network. *Journal of Machine Learning Research*, 2007.



Karim Ali received his Bachelor's (Honours) and Master's degrees from the Electrical and Computer Engineering Department at McGill University. He is currently working towards his PhD degree at the École Polytechnique Fédérale de Lausanne (EPFL). His research interests include statistical learning techniques applied to computer vision and graph-based algorithms.



François Fleuret received the PhD degree in probability from the University of Paris VI in 2000, and the habilitation degree in Applied Mathematics from the University of Paris XIII in 2006. He holds a Senior Researcher position at the Idiap Research Institute in Switzerland. His main research interests are at the interface between statistical methods and algorithmic, centered on the development of algorithmically efficient machine learning techniques.



David Hasler holds the position of deputy section head at the Vision Embedded System Lab in a technology transfer institution (CSEM) where he manages the research of a R&D team of 10 people. He received the MS degree in Micro-engineering (1996) and the Ph.D. in communication systems (2001), both from the Swiss federal institute of technology in Lausanne (EPFL). His research interest range from imaging front-ends to acquire visual information from a scene to classification algorithms for scene interpretation.



Pascal Fua received an engineering degree from Ecole Polytechnique, Paris, in 1984 and the Ph.D. degree in Computer Science from the University of Orsay in 1989. He joined EPFL (Swiss Federal Institute of Technology) in 1996 where he is now a Professor in the School of Computer and Communication Science. His research interests include shape modeling and motion recovery from images, analysis of microscopy images, and Augmented Reality. He has (co)authored over 150 publications in refereed journals and conferences. He has been an associate editor of IEEE journal Transactions for Pattern Analysis and Machine Intelligence and has often been a program committee member, area chair, and program chair of major vision conferences.

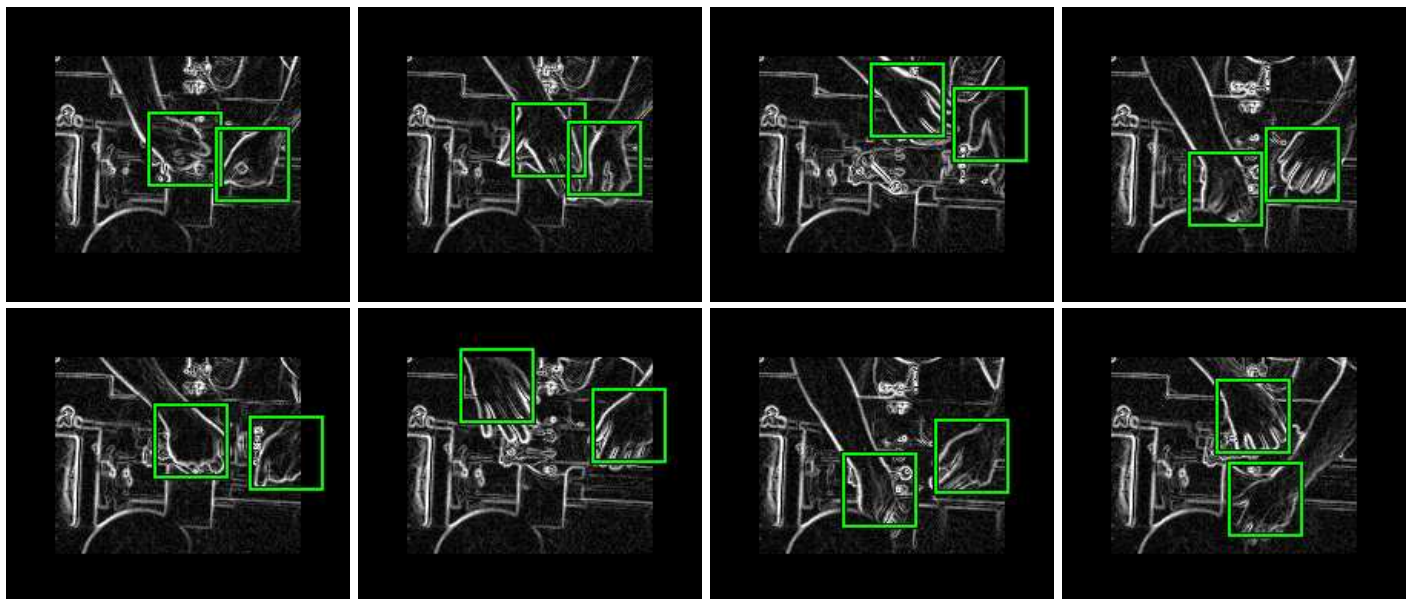


Fig. 17: Some example detections sampled uniformly at random across the entire test set obtained from our hardware-specialized camera. True positive rate is 90%. Correct detections are shown in green whereas false alarms are shown in red. Detection proceeds frame by frame independently with no temporal constraints, not even background subtraction.

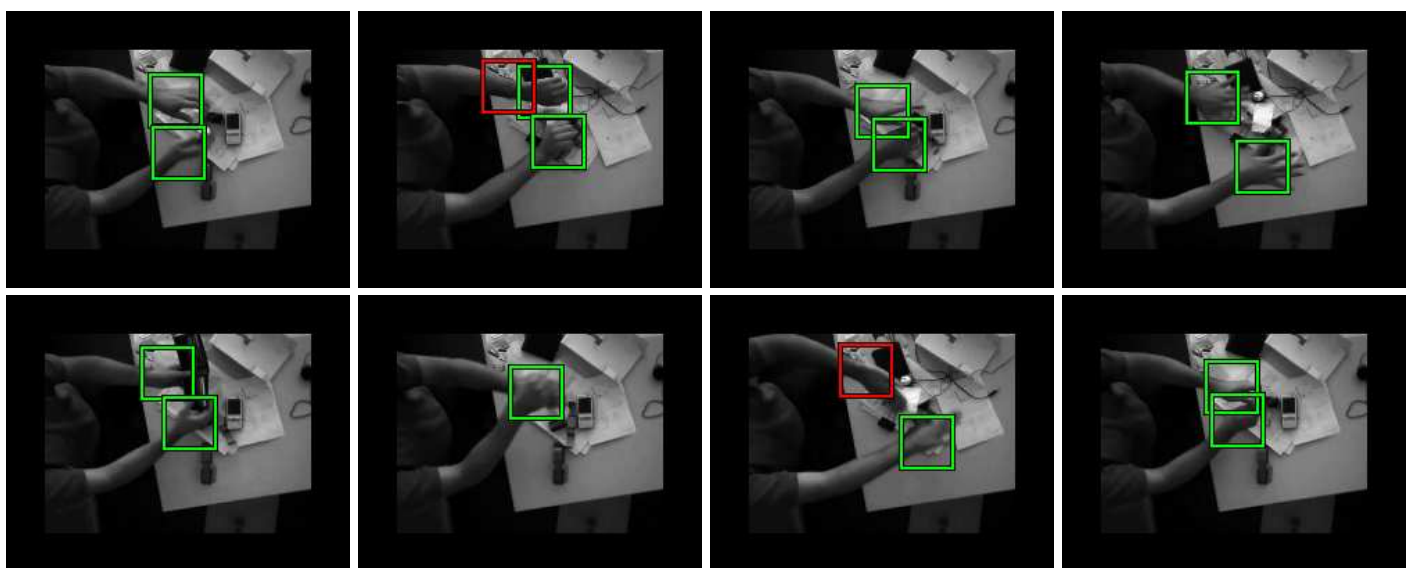


Fig. 18: Some example detections sampled uniformly at random across the entire test set obtained from the webcam. True positive rate is 90%. Correct detections are shown in green whereas false alarms are shown in red. Detection proceeds frame by frame independently with no temporal constraints, not even background subtraction.



Fig. 19: Some examples from our car test set. True positive rate is 85%. Correct detections are shown in green whereas false alarms are shown in red. There are 198,000 tests per image.